#3

Figure 1

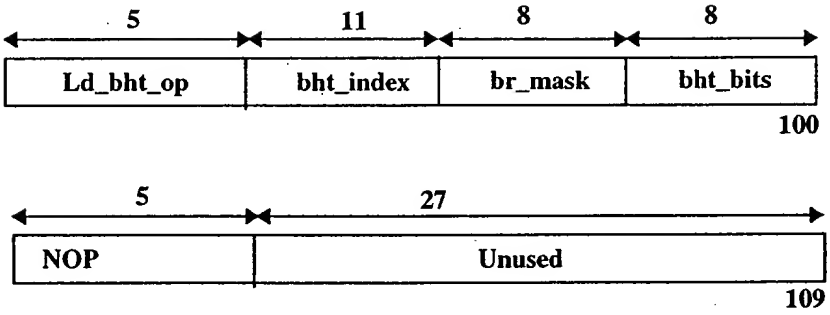

5
11
8
8
Ld_bht_op
bht_index
br_mask
bht_bits
100
5
27
NOP
Unused
109

Figure 2

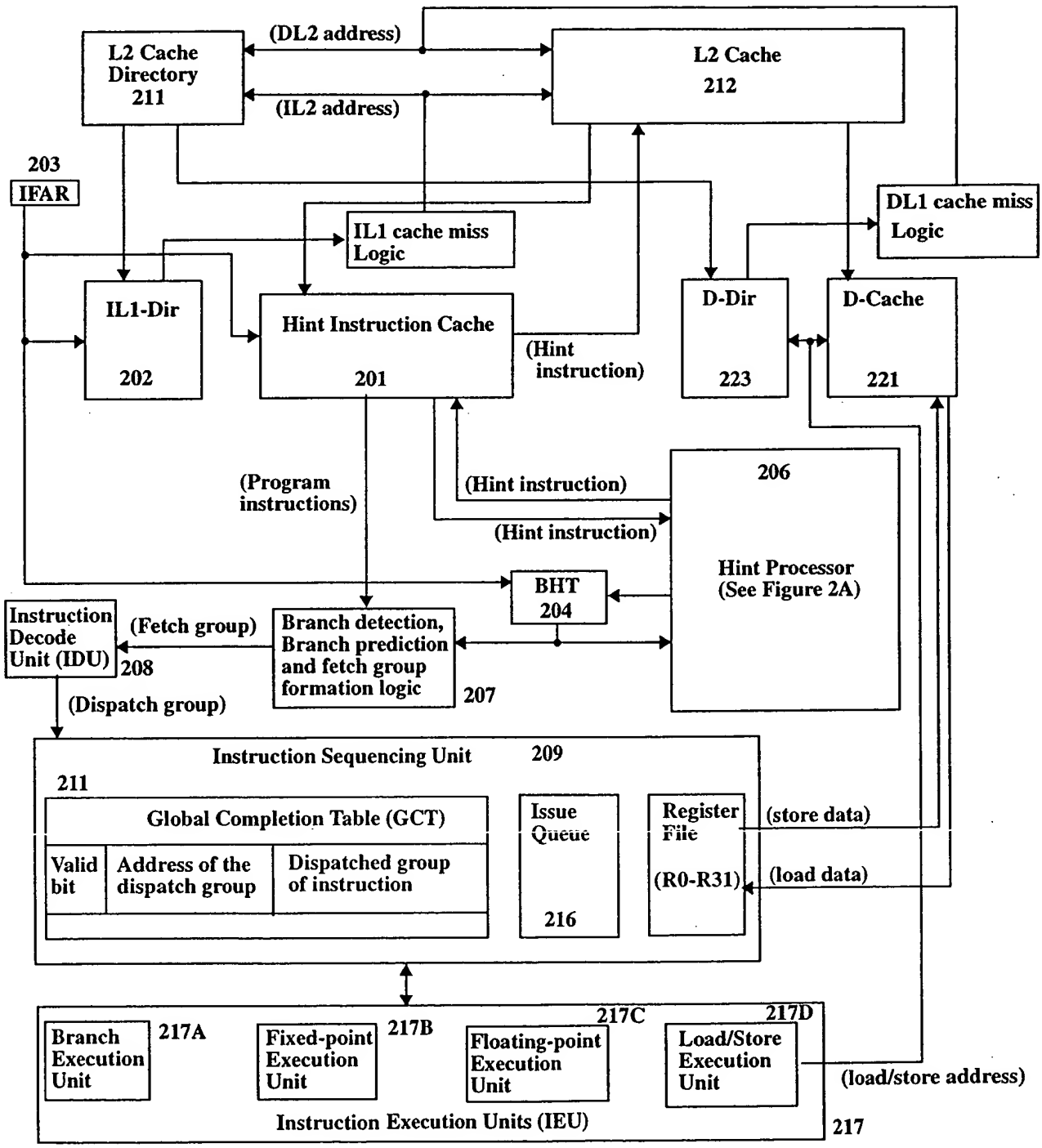

L2 Cache Directory 211
(DL2 address)
(IL2 address)
L2 Cache 212
203
IFAR
IL1 cache miss Logic
DL1 cache miss Logic
IL1-Dir 202
Hint Instruction Cache 201
(Hint instruction)
D-Dir 223
D-Cache 221
(Program instructions)
(Hint instruction)
(Hint instruction)
206
Hint Processor (See Figure 2A)
BHT 204
Instruction Decode Unit (IDU)
208
(Fetch group)
Branch detection, Branch prediction and fetch group formation logic
207
(Dispatch group)
Instruction Sequencing Unit
209
211
Global Completion Table (GCT)
Valid bit
Address of the dispatch group
Dispatched group of instruction
Issue Queue
216
Register File
(R0-R31)
(store data)
(load data)
Branch Execution Unit
217A
Fixed-point Execution Unit
217B
Floating-point Execution Unit
217C
Load/Store Execution Unit
217D
(load/store address)
Instruction Execution Units (IEU)
217

Figure 2A

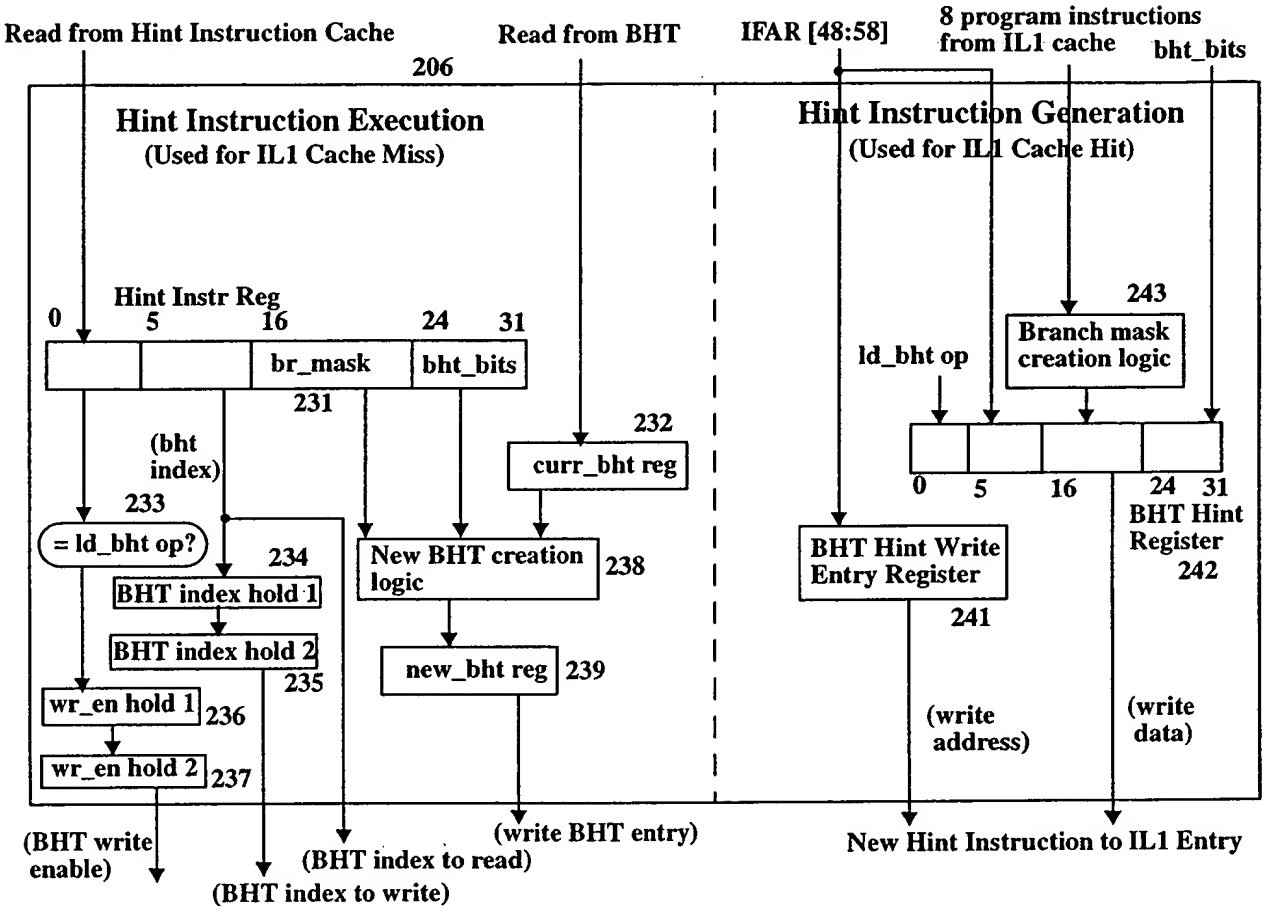


Figure 2B

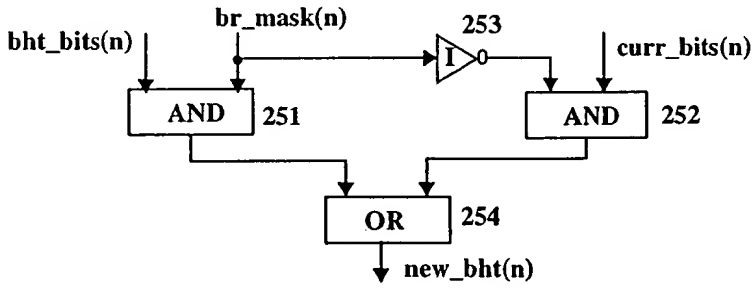


Figure 3

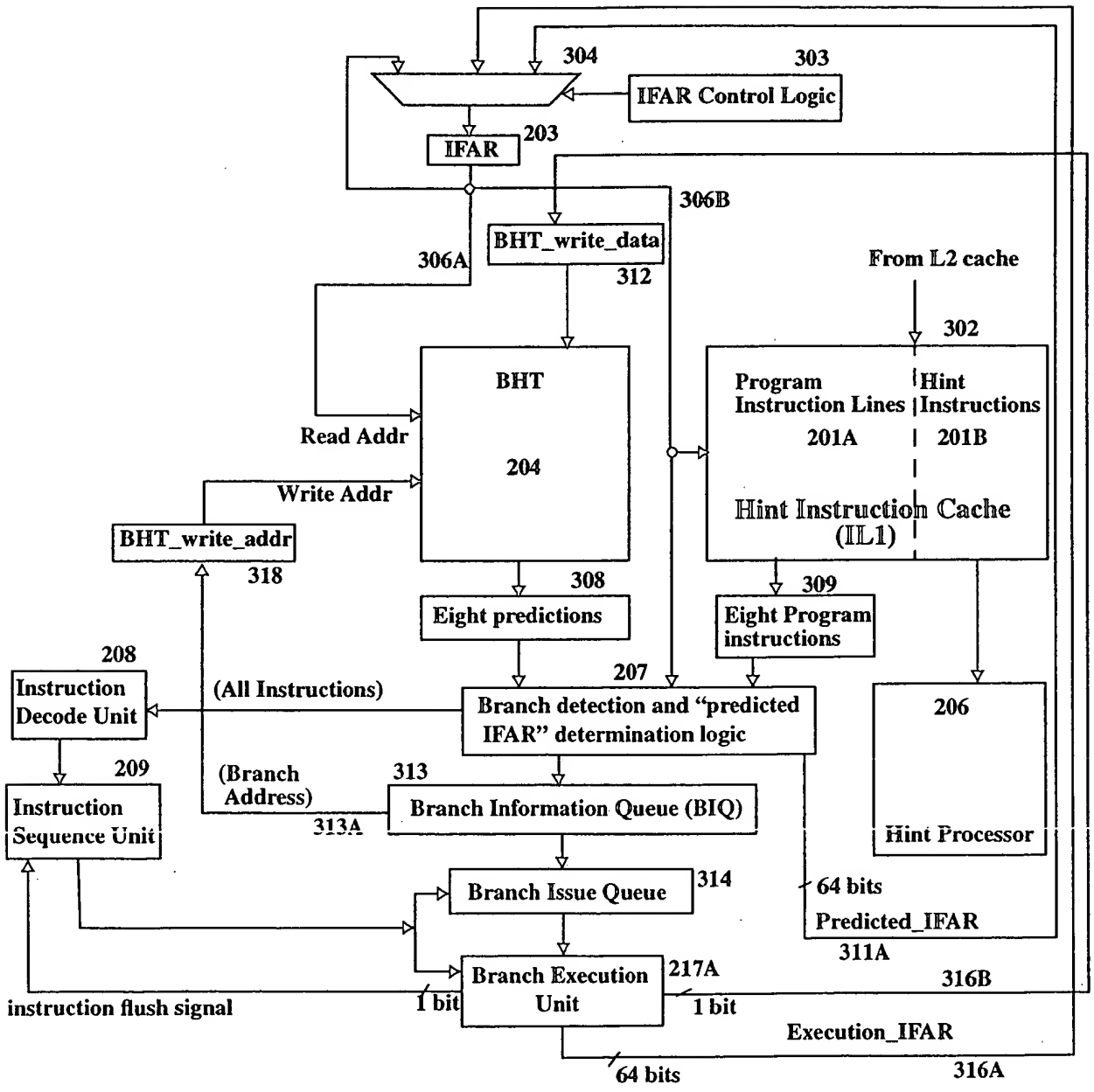
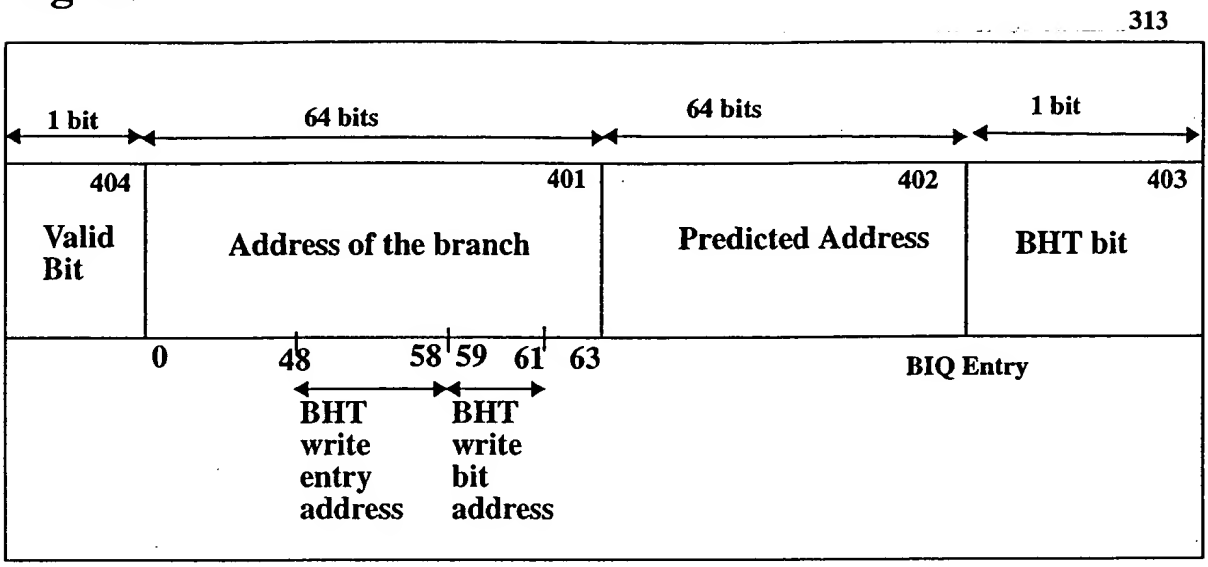
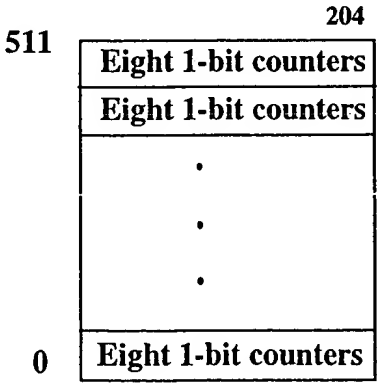

304
303
IFAR Control Logic
IFAR
203
306B
306A
BHT_write_data
312
From L2 cache
302
BHT
204
Program Instruction Lines
201A
Hint Instructions
201B
Hint Instruction Cache (IL1)
Read Addr
Write Addr
BHT_write_addr
318
308
Eight predictions
309
Eight Program instructions
208
Instruction Decode Unit
(All Instructions)
207
Branch detection and "predicted IFAR" determination logic
206
209
(Branch Address)
313
313A
Branch Information Queue (BIQ)
Instruction Sequence Unit
Hint Processor
Branch Issue Queue
314
64 bits
Predicted_IFAR
311A
Branch Execution Unit
217A
316B
instruction flush signal
1 bit
1 bit
Execution_IFAR
64 bits
316A

Figure 4



Branch Information Queue (BIQ)

Figure 5



BHT

Figure 6

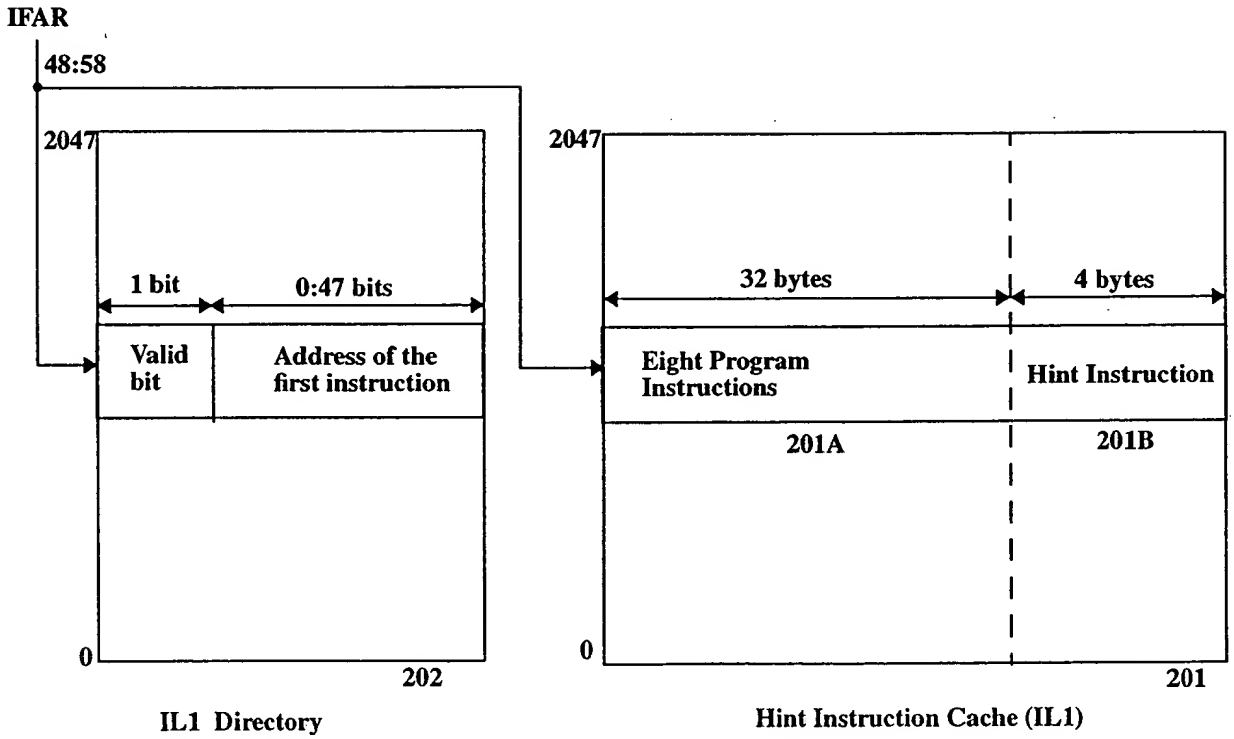

IFAR
48:58
2047
1 bit
0:47 bits
Valid bit
Address of the first instruction
0
202
IL1 Directory
2047
32 bytes
4 bytes
Eight Program Instructions
Hint Instruction
201A
201B
0
201
Hint Instruction Cache (IL1)

Figure 7

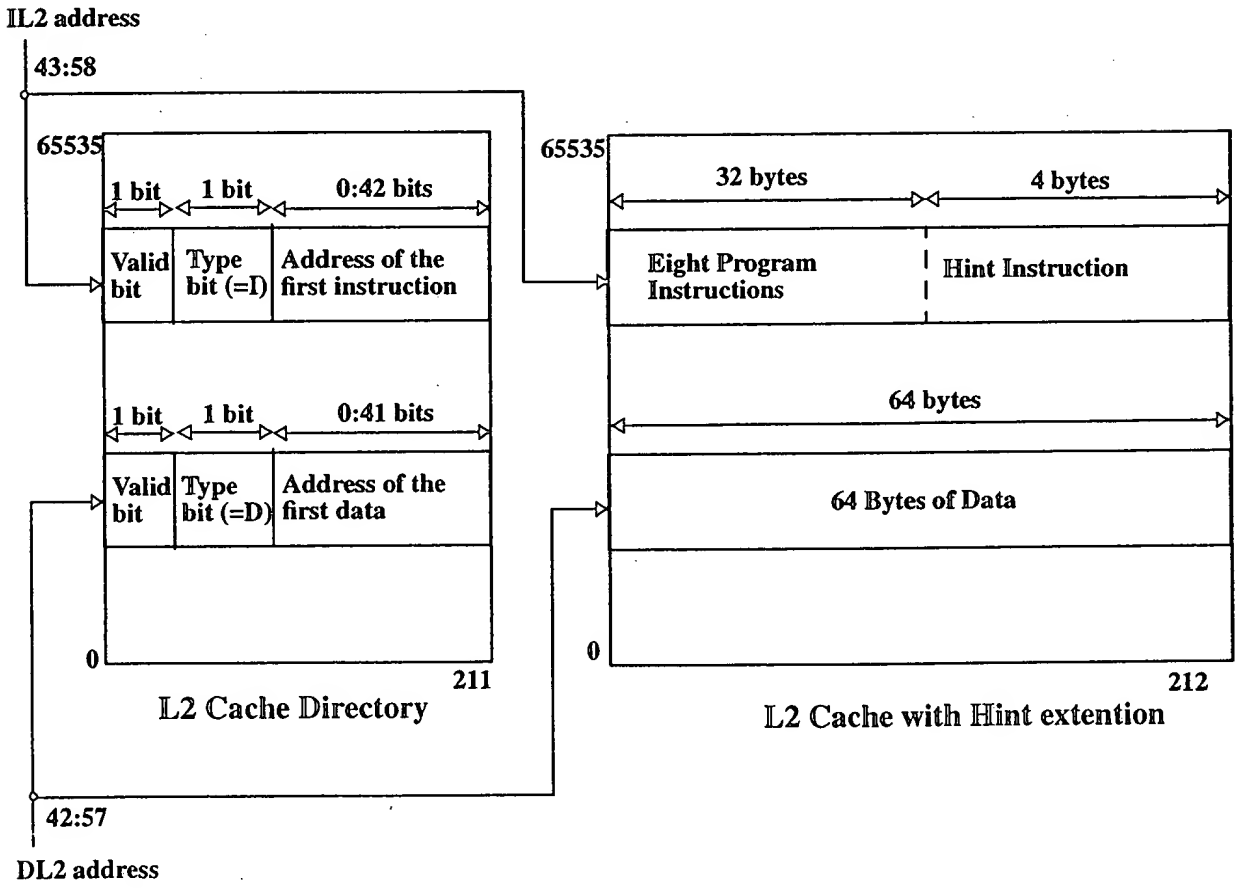

IL2 address
43:58
65535
1 bit
1 bit
0:42 bits
Valid bit
Type bit (=I)
Address of the first instruction
1 bit
1 bit
0:41 bits
Valid bit
Type bit (=D)
Address of the first data
0
211
L2 Cache Directory
42:57
DL2 address
65535
32 bytes
4 bytes
Eight Program Instructions
Hint Instruction
64 bytes
64 Bytes of Data
0
212
L2 Cache with Hint extention

Figure 8

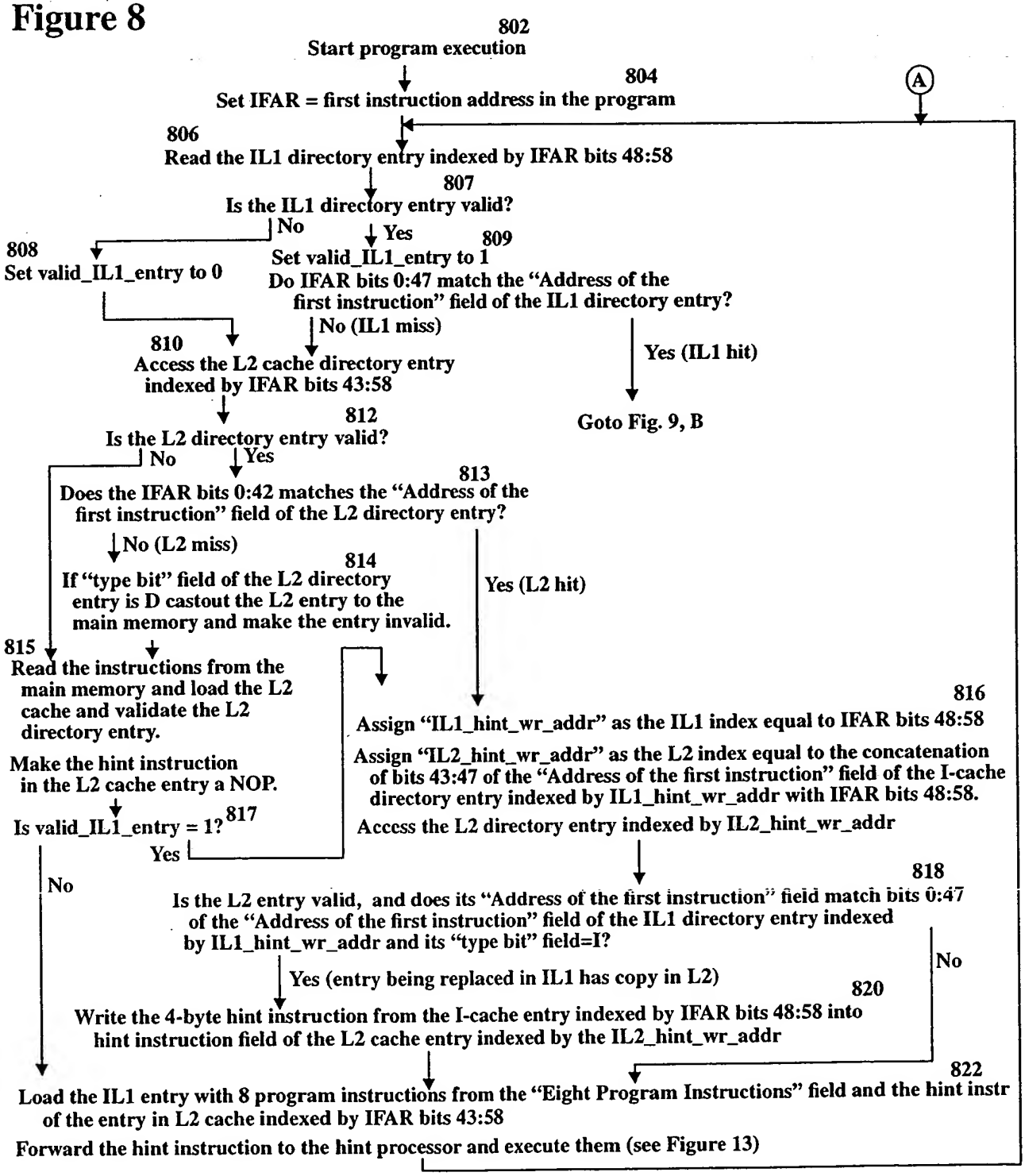

802
Start program execution
804
Set IFAR = first instruction address in the program
A
806
Read the IL1 directory entry indexed by IFAR bits 48:58
807
Is the IL1 directory entry valid?
No
Yes
808
Set valid_IL1_entry to 0
809
Set valid_IL1_entry to 1
Do IFAR bits 0:47 match the "Address of the first instruction" field of the IL1 directory entry?
No (IL1 miss)
Yes (IL1 hit)
810
Access the L2 cache directory entry indexed by IFAR bits 43:58
Goto Fig. 9, B
812
Is the L2 directory entry valid?
No
Yes
813
Does the IFAR bits 0:42 matches the "Address of the first instruction" field of the L2 directory entry?
No (L2 miss)
814
If "type bit" field of the L2 directory entry is D castout the L2 entry to the main memory and make the entry invalid.
Yes (L2 hit)
815
Read the instructions from the main memory and load the L2 cache and validate the L2 directory entry.
Make the hint instruction in the L2 cache entry a NOP.
Is valid_IL1_entry = 1?
817
Yes
No
816
Assign "IL1_hint_wr_addr" as the IL1 index equal to IFAR bits 48:58
Assign "IL2_hint_wr_addr" as the L2 index equal to the concatenation of bits 43:47 of the "Address of the first instruction" field of the I-cache directory entry indexed by IL1_hint_wr_addr with IFAR bits 48:58.
Access the L2 directory entry indexed by IL2_hint_wr_addr
818
Is the L2 entry valid, and does its "Address of the first instruction" field match bits 0:47 of the "Address of the first instruction" field of the IL1 directory entry indexed by IL1_hint_wr_addr and its "type bit" field=I?
No
Yes (entry being replaced in IL1 has copy in L2)
820
Write the 4-byte hint instruction from the I-cache entry indexed by IFAR bits 48:58 into hint instruction field of the L2 cache entry indexed by the IL2_hint_wr_addr
822
Load the IL1 entry with 8 program instructions from the "Eight Program Instructions" field and the hint instr of the entry in L2 cache indexed by IFAR bits 43:58
Forward the hint instruction to the hint processor and execute them (see Figure 13)

Figure 9

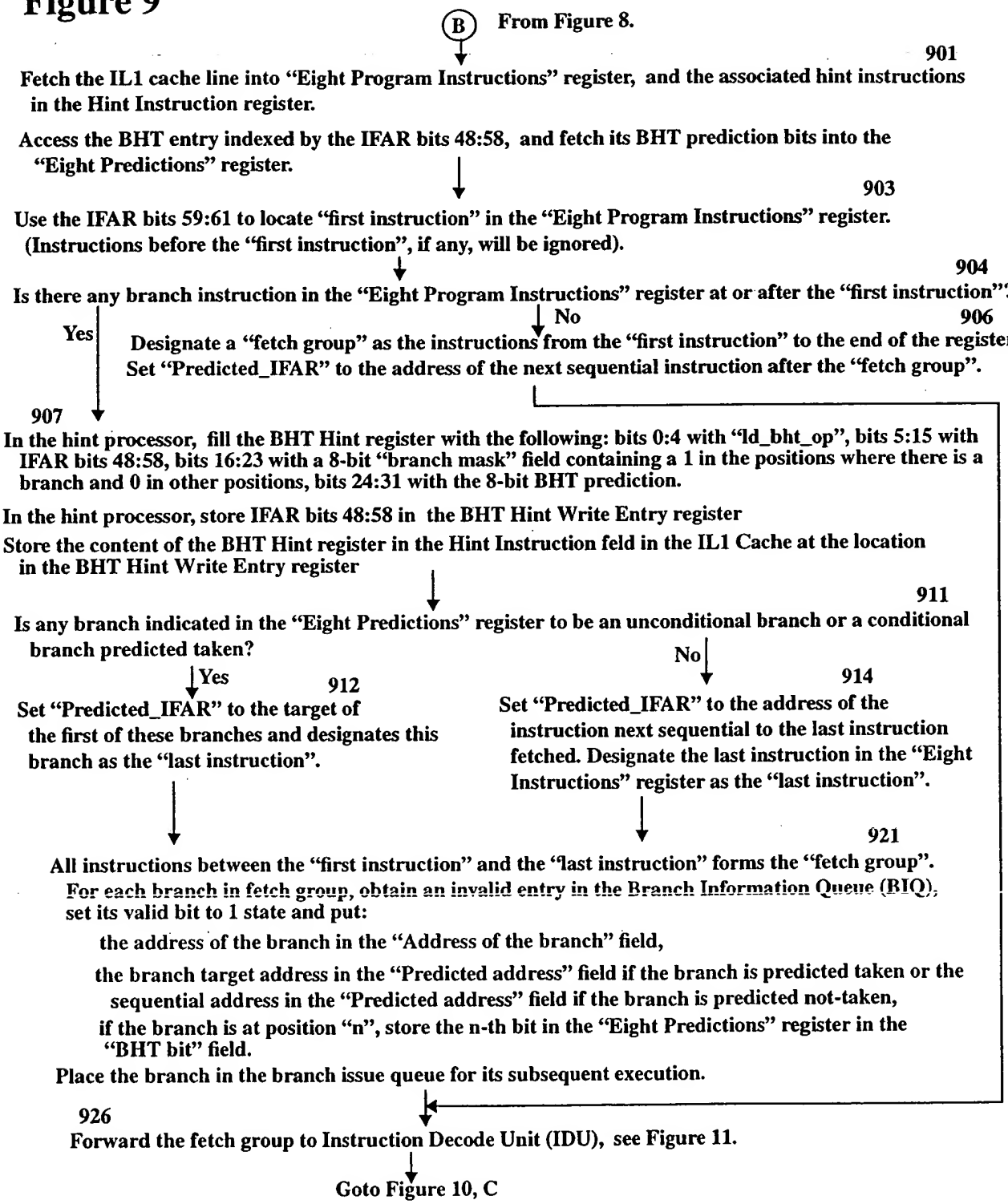

B
From Figure 8.
901
Fetch the IL1 cache line into "Eight Program Instructions" register, and the associated hint instructions in the Hint Instruction register.
Access the BHT entry indexed by the IFAR bits 48:58, and fetch its BHT prediction bits into the "Eight Predictions" register.
903
Use the IFAR bits 59:61 to locate "first instruction" in the "Eight Program Instructions" register. (Instructions before the "first instruction", if any, will be ignored).
904
Is there any branch instruction in the "Eight Program Instructions" register at or after the "first instruction"?
Yes
No
906
Designate a "fetch group" as the instructions from the "first instruction" to the end of the register.
Set "Predicted_IFAR" to the address of the next sequential instruction after the "fetch group".
907
In the hint processor, fill the BHT Hint register with the following: bits 0:4 with "ld_bht_op", bits 5:15 with IFAR bits 48:58, bits 16:23 with a 8-bit "branch mask" field containing a 1 in the positions where there is a branch and 0 in other positions, bits 24:31 with the 8-bit BHT prediction.
In the hint processor, store IFAR bits 48:58 in the BHT Hint Write Entry register
Store the content of the BHT Hint register in the Hint Instruction feld in the IL1 Cache at the location in the BHT Hint Write Entry register
911
Is any branch indicated in the "Eight Predictions" register to be an unconditional branch or a conditional branch predicted taken?
Yes
912
Set "Predicted_IFAR" to the target of the first of these branches and designates this branch as the "last instruction".
No
914
Set "Predicted_IFAR" to the address of the instruction next sequential to the last instruction fetched. Designate the last instruction in the "Eight Instructions" register as the "last instruction".
921
All instructions between the "first instruction" and the "last instruction" forms the "fetch group".
For each branch in fetch group, obtain an invalid entry in the Branch Information Queue (BIQ), set its valid bit to 1 state and put:
the address of the branch in the "Address of the branch" field,
the branch target address in the "Predicted address" field if the branch is predicted taken or the sequential address in the "Predicted address" field if the branch is predicted not-taken,
if the branch is at position "n", store the n-th bit in the "Eight Predictions" register in the "BHT bit" field.
Place the branch in the branch issue queue for its subsequent execution.
926
Forward the fetch group to Instruction Decode Unit (IDU), see Figure 11.
Goto Figure 10, C

Figure 10

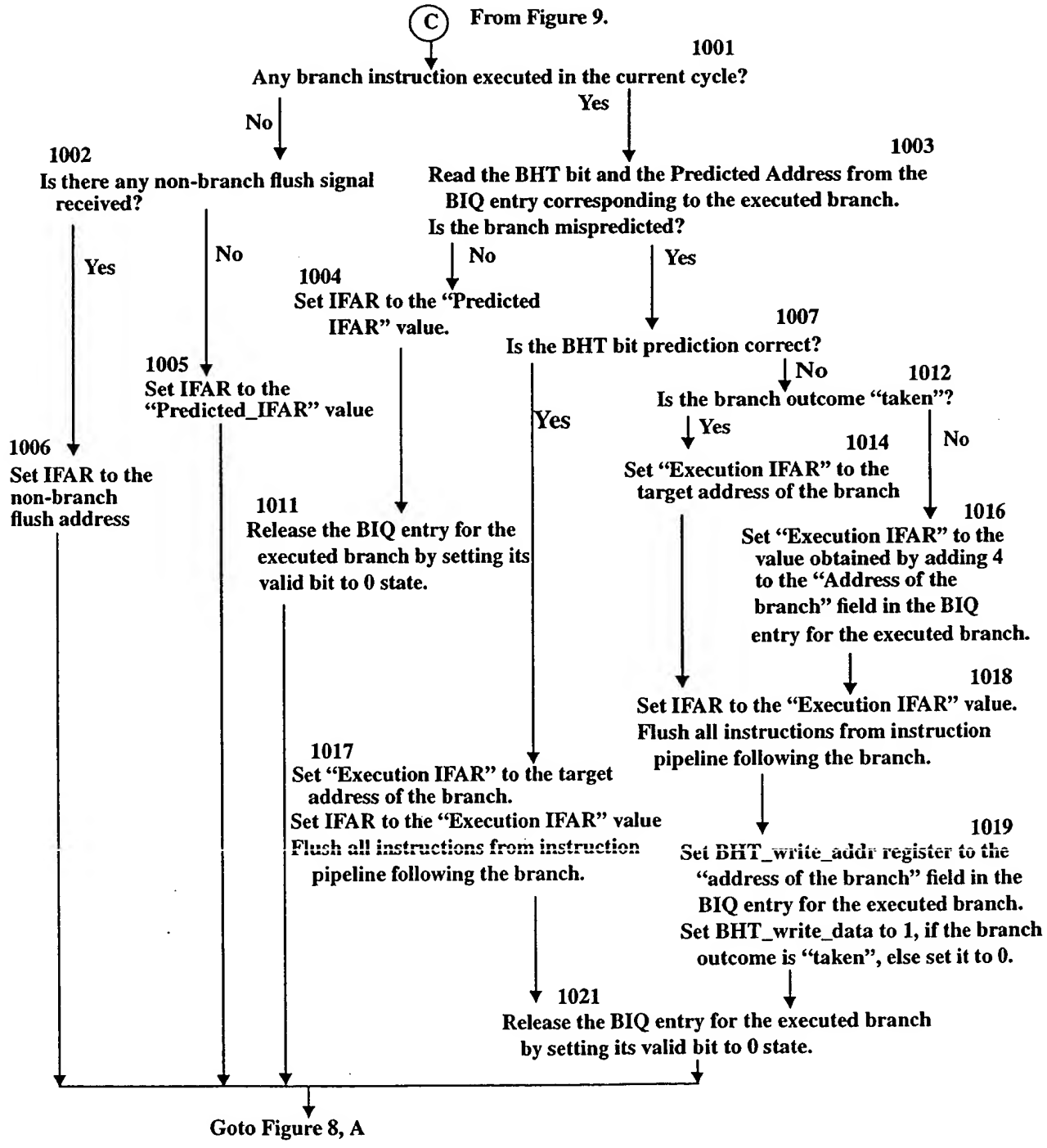

C
From Figure 9.
1001
Any branch instruction executed in the current cycle?
No
Yes
1002
Is there any non-branch flush signal received?
1003
Read the BHT bit and the Predicted Address from the BIQ entry corresponding to the executed branch.
Is the branch mispredicted?
Yes
No
Yes
No
1004
Set IFAR to the "Predicted IFAR" value.
1005
Set IFAR to the "Predicted_IFAR" value
1006
Set IFAR to the non-branch flush address
1007
Is the BHT bit prediction correct?
No
Yes
1011
Release the BIQ entry for the executed branch by setting its valid bit to 0 state.
1012
Is the branch outcome "taken"?
Yes
No
1014
Set "Execution IFAR" to the target address of the branch
1016
Set "Execution IFAR" to the value obtained by adding 4 to the "Address of the branch" field in the BIQ entry for the executed branch.
1017
Set "Execution IFAR" to the target address of the branch.
Set IFAR to the "Execution IFAR" value
Flush all instructions from instruction pipeline following the branch.
1018
Set IFAR to the "Execution IFAR" value.
Flush all instructions from instruction pipeline following the branch.
1019
Set BHT_write_addr register to the "address of the branch" field in the BIQ entry for the executed branch.
Set BHT_write_data to 1, if the branch outcome is "taken", else set it to 0.
1021
Release the BIQ entry for the executed branch by setting its valid bit to 0 state.
Goto Figure 8, A

Figure 11

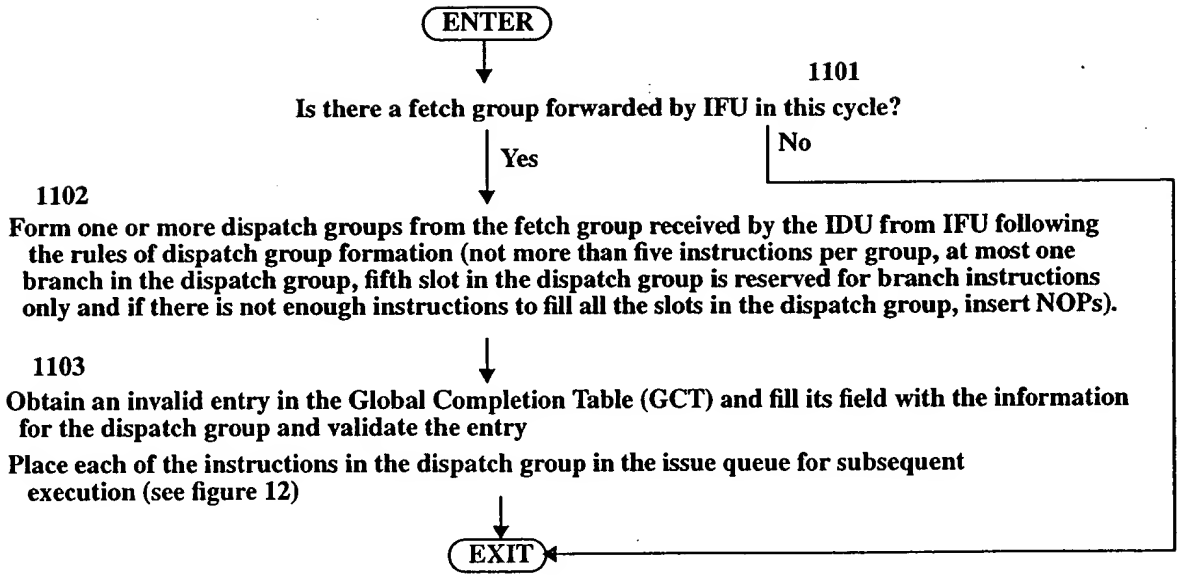

ENTER
1101
Is there a fetch group forwarded by IFU in this cycle?
Yes
No
1102
Form one or more dispatch groups from the fetch group received by the IDU from IFU following the rules of dispatch group formation (not more than five instructions per group, at most one branch in the dispatch group, fifth slot in the dispatch group is reserved for branch instructions only and if there is not enough instructions to fill all the slots in the dispatch group, insert NOPs).
1103
Obtain an invalid entry in the Global Completion Table (GCT) and fill its field with the information for the dispatch group and validate the entry
Place each of the instructions in the dispatch group in the issue queue for subsequent execution (see figure 12)
EXIT

Figure 12

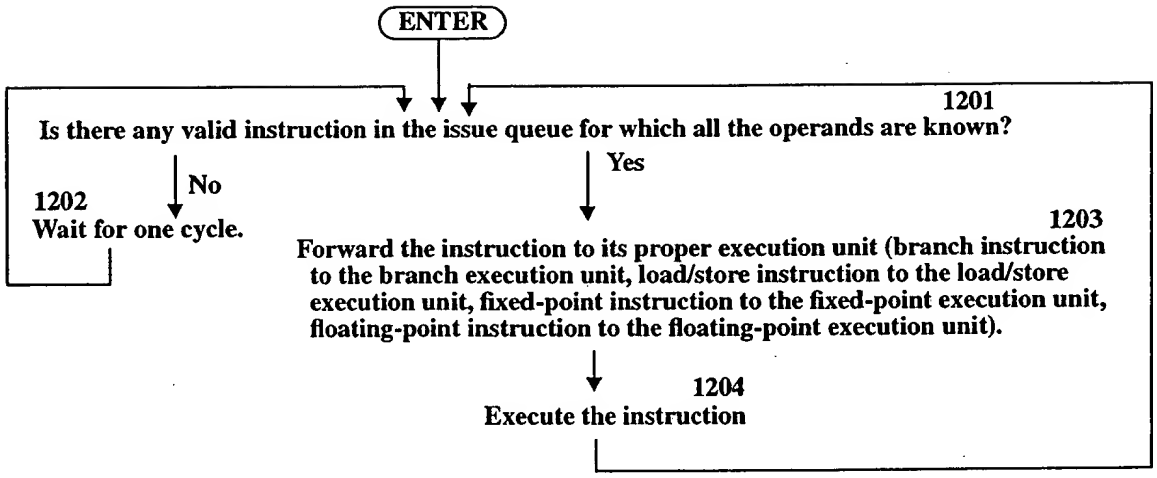
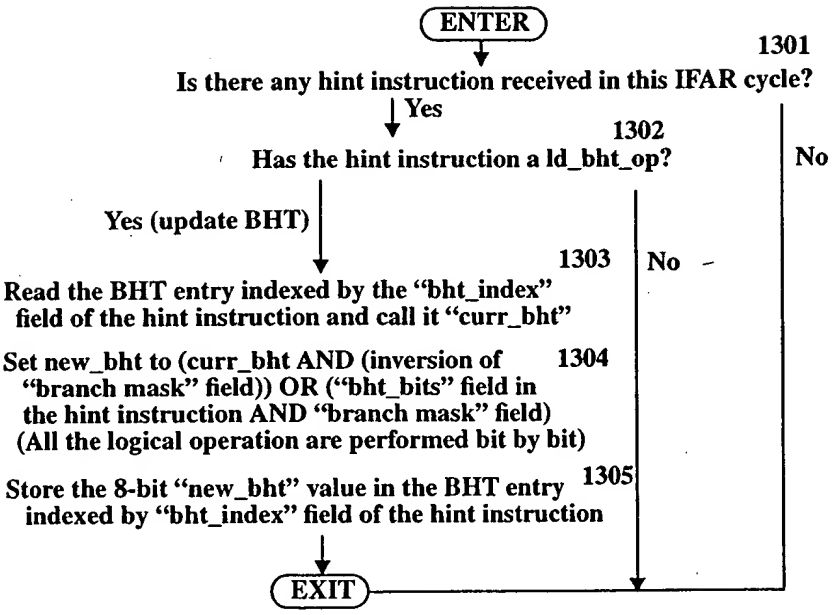

ENTER
1201
Is there any valid instruction in the issue queue for which all the operands are known?
No
Yes
1202
Wait for one cycle.
1203
Forward the instruction to its proper execution unit (branch instruction to the branch execution unit, load/store instruction to the load/store execution unit, fixed-point instruction to the fixed-point execution unit, floating-point instruction to the floating-point execution unit).
1204
Execute the instruction

Figure 13


ENTER
1301
Is there any hint instruction received in this IFAR cycle?
Yes
No
1302
Has the hint instruction a ld_bht_op?
Yes (update BHT)
No
1303
Read the BHT entry indexed by the "bht_index" field of the hint instruction and call it "curr_bht"
Set new_bht to (curr_bht AND (inversion of "branch mask" field)) OR ("bht_bits" field in the hint instruction AND "branch mask" field) (All the logical operation are performed bit by bit)
1304
Store the 8-bit "new_bht" value in the BHT entry indexed by "bht_index" field of the hint instruction
1305
EXIT